Please replace the paragraph beginning at page 12, line 8 with the following rewritten paragraph:

--Each of the modules is associated with an enabling condition, which is a condition which determines whether the module will be evaluated for a given input object. Enabling conditions can refer to attribute values, attribute exception values, attribute states (e.g., whether the attribute has a value or whether an exception occurred when attempting to evaluate it), module states and module exception values. The enabling conditions are graphically represented as broken line hexagons 211, 221, 231, 241, 251, 261. Enabling conditions 211, 251, and 261 contain TRUE, which will always evaluate to a true condition, and therefore the Identify_Caller module 210, Routing_Decisions module 250, and Calculate_Wrap_Up module 260 will be evaluated for each input object. Enabling condition 221 contains the expression: VAL (ACCOUNT_NUMBER). The function VAL (X) will return a true condition if the attribute X is in the state VALUE, otherwise, false will be returned. Therefore, the enabling condition 221 indicates that the Info_About_Customer module 220 will be evaluated if the attribute ACCOUNT-NUMBER is in the state VALUE. If the attribute ACCOUNT-NUMBER is in state EXCEPTION, DISABLED, or FAILED, then enabling condition 221 will evaluate to false and the Info_About_Customer module 220 will not be evaluated. If the attribute ACCOUNT_NUMBER is in state UNINITIALIZED, then enabling condition 221 cannot yet be evaluated. Thus, the evaluation of enabling condition 221 depends on the attribute ACCOUNT-NUMBER first receiving a state other than UNINITIALIZED. It is noted that this dependency is implicit in the DL specification and not explicitly specified by the system designer or programmer.--

Please replace the paragraph beginning at page 13, line 9 with the following rewritten paragraph:

--As shown in Fig. 4, the module name is specified in line 1, and an indication of which module the current module is a sub-module of is given in line 2. The next section defines the input attributes (line 3). The next section defines the output attributes (lines 4 – 14). Line 15 specifies the enabling condition, which corresponds to the enabling condition 211 shown in Fig. 2. The type of the module, in this case flowchart, is specified on line 16. The computation section of the textual specification (line 17) indicates how attributes are to be evaluated. For this module, the attributes will be evaluated according to the flowchart shown in Fig. 3. Of course, one skilled in the art could convert the flowchart of Fig. 3 to program code to implement the logic flow shown in Fig. 3. Such code is not included in Fig. 4 because it is not necessary for an understanding of the principles of the present invention. Finally, line 18 indicates that this module has a side-effect. The side-effect action is an interactive voice response (IVR) unit dip (line 19).--

A3

Please replace the paragraph beginning at page 15, line 4 with the following rewritten paragraph:

--The DL specification further defining the Info_About_Customer module 220 (Fig. 2) is shown graphically in Fig. 5 and textually in Fig. 6. This Info_About_Customer module 220 is a declarative module and is therefore further defined in terms of sub-modules. The Get_Recent_Contacts_For_This_Customer module 504, the Get_Recent_Purchases_For_This_Customer module 508, the Get_Account_History_For_This_Customer module 512, and the Calculate_Cust_Value module 528 will always be evaluated because their respective enabling conditions 502, 506, 510, 526 are always true. It is noted that the graphical representation of these modules indicate that they are foreign modules. Each of these modules performs an external database retrieval function. If the attribute RECENT_CONTACTS has a state of VALUE, then the enabling condition 514 will be True and the

A4